Comparison of the FPGA Implementation of Two Multilevel Space Vector PWM Algorithms

Óscar López, *Member, IEEE*, Jacobo Álvarez, Jesús Doval-Gandoy, *Member, IEEE*, Francisco D. Freijedo, *Member, IEEE*, Andrés Nogueiras, *Member, IEEE*, Alfonso Lago, *Member, IEEE*, and Carlos M. Peñalver, *Member, IEEE*

***Abstract*—Multilevel converters can meet the increasing demand of power ratings and power quality associated with reduced harmonic distortion and lower electromagnetic interference. When the number of levels increases, it is necessary to control more and more switches in parallel. Field programmable gate arrays (FPGAs), with their concurrent processing capability, are suitable for the implementation of multilevel modulation algorithms. Among them, space vector pulsewidth modulation algorithms offer great flexibility to optimize switching waveforms and are well suited for digital implementation. In this paper, two algorithms, 2-D and 3-D, are analyzed and implemented in an FPGA. In order to carry out the implementation, both algorithms have been described in very high speed integrated circuit hardware description language, partly hand coded, and partly automatically generated using the System Generator tool. Both implementations are compared in terms of implementation complexity and logic resources required. Finally, test results with a neutral-point-clamped inverter are presented.**

***Index Terms*—Field programmable gate array (FPGA), multilevel inverter, space vector pulsewidth modulation (SVPWM).**

I. Introduction

THE TERM multilevel starts with the three-level inverter introduced by Nabae *et al.* [1]. Nowadays, multilevel converters are becoming increasingly popular in power applications, owing to their ability to meet the increasing demand of power ratings and power quality associated with reduced harmonic distortion and lower electromagnetic interference [2]. Multilevel topologies have been used as rectifiers in [3] and [4] for the connection of renewable energy sources to the utility grid in [5] and as a power-conditioning system of variable speed wind turbines in [6].

The combination of a large number of semiconductors to achieve high power ratings is well established. Choosing an arrangement where all devices are individually controlled, rather than switched together in series, provides more control opportunities. Voltage-source multilevel-inverter topologies [1], [7]–[10] synthesize a voltage waveform from several voltage levels typically obtained from several capacitors or dc sources. The commutation of the power switches permits the addition of the voltages of these sources, which reach high voltage at the output, whereas the power semiconductors must withstand only reduced voltages.

By increasing the number of levels in the inverter, the output voltages have more steps in generating a staircase waveform, which has a reduced harmonic distortion. However, a larger number of levels increase the number of devices that must be controlled and the control complexity. At present, there are no commercial digital signal processors (DSPs) having appropriate built-in pulsewidth modulation (PWM) units that are enough to control the large amount of switches used by multilevel converters. A software implementation of these PWM units is very time-consuming. Therefore, a fast and expensive DSP is needed to carry out the modulation and the control processes. An architecture where one field programmable gate array (FPGA) carries out the modulation task and one DSP implements the control strategy is better suited for multilevel converters. Thus, cheap DSP and FPGA are needed.

FPGAs' development reached a level of maturity that made them the choice of implementation in many fields [11]. Recent applications of FPGAs in industrial electronics include mobile-robot path planning and intelligent transportation [12], [13], current controlling applied to power converters [14]–[16], real-time hardware-in-the-loop testing for control design [17], controller implementation [18]–[21], separating and recovering independent source signals [22], and neural computation [23]. Digital controllers using a microprocessor, together with an FPGA, have been demonstrated in previous works [24]–[29].

Since the concept of multilevel PWM converter was introduced, various modulation strategies have been developed and studied in detail, such as multilevel sinusoidal PWM, multilevel selective harmonic elimination, and space vector modulation. Among these strategies, the space vector PWM (SVPWM) [30]–[33] stands out because it offers significant flexibility to optimize switching waveforms and is well suited for digital implementation. Complexity and computational cost of traditional SVPWM techniques increase with the number of levels of the converter, and most of all use trigonometric functions or precomputed tables [34]–[37]. The 2-D SVPWM algorithm proposed by Celanovic and Boroyevich [38] and the 3-D one proposed by Prats *et al.* [39] calculate the switching vectors and the times without using angles, trigonometric functions, or precomputed tables. In addition, the complexity and the computational cost are very low. These efficient modulation

Manuscript received June 30, 2006; revised August 22, 2007. This work was supported by the Spanish Ministry of Science and Technology under Project ENE2006-02930.

The authors are with the Department of Electronic Technology, University of Vigo, 36310 Vigo, Spain (e-mail: olopez@uvigo.es).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIE.2008.917159

algorithms are very useful to real-time computation of the switching sequence and the ON-state durations of the respective switching state vectors corresponding to the modulation of multilevel inverters. Moreover, the numeric evaluation of the n-state durations is reduced to a simple addition, and both methods use the minimum number of possible comparators.

The implementation in an FPGA of a SVPWM control for a two-level inverter has been described in detail in [40] and [41]. An FPGA was also used in [42] to control a three-level neutral-point-clamped (NPC) inverter in wind turbine applications. In this paper, two multilevel SVPWM algorithms are analyzed and implemented into an FPGA. A detailed description of the very high speed integrated circuit hardware description language (VHDL) implementation realized is given. To develop all the tasks involved, Matlab Simulink and System Generator for Simulink tools are used [43], [44] as well as Foundation ISE tools [45]. Both implementations have been compared in terms of implementation complexity and resources of the FPGA used. Finally, both implementations have been tested with an NPC inverter.

This paper is organized as follows. Section II analyzes and compares both SVPWM algorithms. In Section III, the algorithms are organized in functional blocks. Section IV describes the hardware implementation of each functional block. Section V shows the architecture of the experimental setup and gives some testing results. Section VI is the conclusion.

II. SVPWM Algorithms

Algorithms proposed by Celanovic and Boroyevich [38] and Prats *et al.* [39] are two SVPWM algorithms which approximate the reference vector from a sequence of vectors selected into the limited set of vectors which can be generated by the inverter. Both algorithms can be applied to inverters with any number of levels in which the output voltage vector $\mathbf{V}_s$ can be written as

$$\mathbf{V}_s = \mathbf{v}_s V_{\mathrm{dc}} / \mathbf{v}_s \in \mathbb{N}^3 \tag{1}$$

where V_{dc} is the output voltage step and $\mathbf{v}_s$ is the space vector corresponding to the output vector $\mathbf{V}_s$. This expression is fulfilled for the most usual multilevel topologies, such as the diode clamped converter [1], [7], the flying-capacitor converter [8], and the cascaded full-bridge converter [46].

The 2-D algorithm guarantees that the line-to-line voltage reference is equal to the averaged line-to-line converter voltage output

$$\begin{pmatrix} v_{ab} \\ v_{bc} \\ v_{ca} \end{pmatrix}_{\mathrm{ref}} = \overline{\begin{pmatrix} v_{ab} \\ v_{bc} \\ v_{ca} \end{pmatrix}}. \tag{2}$$

The homopolar component of the reference is not taken into account; therefore, the line-to-neutral voltage of the reference and the output can be different. Hence, this algorithm is only useful for inverters feeding a load without neutral or with floating neutral.

The 3-D algorithm assures that the line-to-neutral voltage reference and the averaged line-to-neutral converter voltage output are equal

$$\begin{pmatrix} v_a \\ v_b \\ v_c \end{pmatrix}_{\mathrm{ref}} = \overline{\begin{pmatrix} v_a \\ v_b \\ v_c \end{pmatrix}}. \tag{3}$$

This algorithm takes into account the homopolar component of the reference; therefore, it is well suited for inverters connected to a load with no floating neutral. However, it can be used with the inverter feeding the same loads of the 2-D algorithm, but wasting the capabilities of the dc bus, because the algorithm does not inject the proper homopolar component which allows extending the modulation index.

The 2-D algorithm takes its name from the fact that vectors with a different nonhomopolar component can be represented in a 2-D space. The 2-D algorithm implemented in this paper uses the following transformation [38]:

$$\begin{pmatrix} v_g \\ v_h \end{pmatrix} = \frac{1}{3} \begin{pmatrix} 2 & -1 & -1 \\ -1 & 2 & -1 \end{pmatrix} \begin{pmatrix} v_{ab} \\ v_{bc} \\ v_{ca} \end{pmatrix}. \tag{4}$$

The set of output vectors that the converter is able to generate inside the control space determines the region where the reference vector can be synthesized. When the reference vector lies outside this control region, it is in the overmodulation zone, and it cannot be approximated accurately. Fig. 1 shows these control regions of both algorithms: a hexagon in the 2-D case and a hexahedron in the 3-D case.

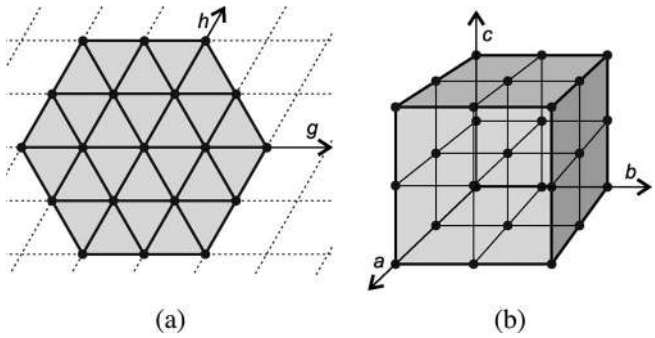

Fig. 1. Control regions. (a) 2-D algorithm. (b) 3-D algorithm.

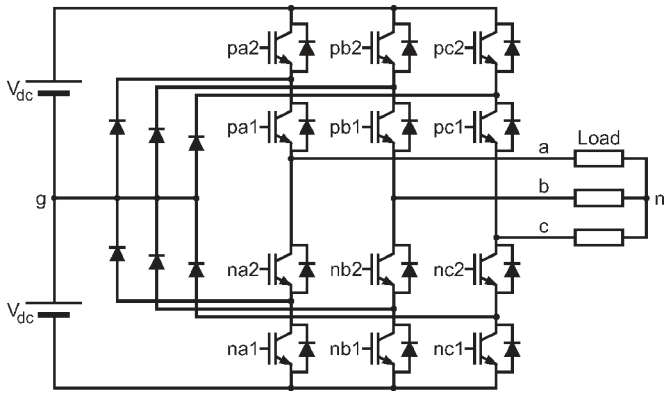

Fig. 2. Three-phase NPC inverter.

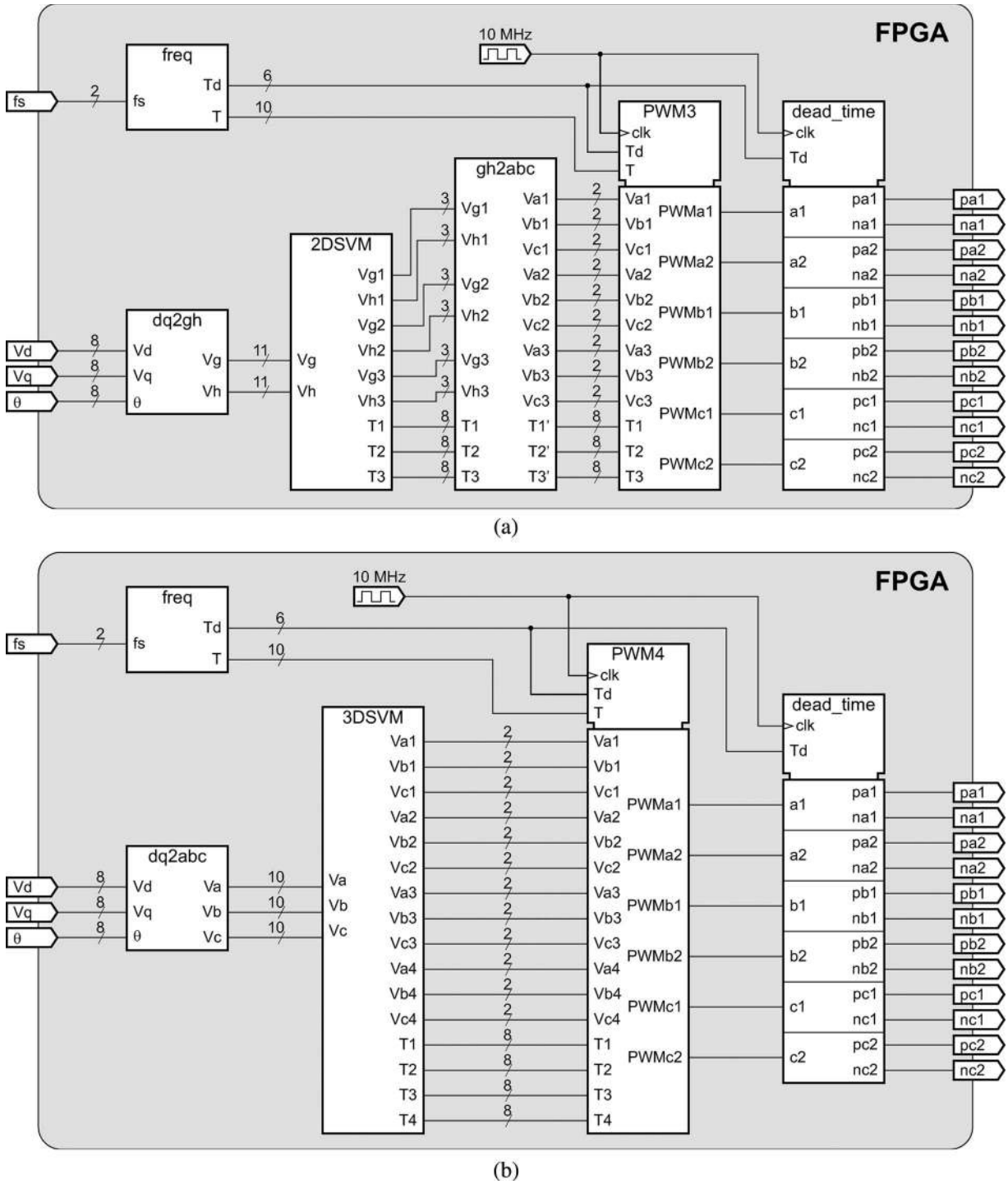

Fig. 3. Implementation diagrams of the SVPWM algorithms. (a) 2-D SVPWM algorithm. (b) 3-D SVPWM algorithm.

Since both space vector algorithms try to approximate a reference vector from a limited set of vectors inside the control region, they follow the same main steps.

1) Place the reference vector in the control space.
2) Search the nearest generating vectors to the reference vector: three vectors in the 2-D algorithm and four ones in the 3-D algorithm.
3) Calculate the on-time of each generating vector.
4) Translate the generating vectors in switching signals.
5) Add the dead times to the switching signals.

All these steps must be repeated each switching cycle.

The 3-D modulation problem has a unique solution, and the 3-D algorithm provides this solution with sorted vectors so that the switching number between two consecutive vectors is minimized. The 2-D algorithm does not have this characteristic because there are several line-to-neutral voltage combinations that provide the same line-to-line voltage output. That is, each 2-D vector in the gh coordinate system corresponds to several

generating vectors in the abc coordinate system. Relationship between both frames is given by [38]

$$\begin{pmatrix} v_a \\ v_b \\ v_c \end{pmatrix} = \begin{pmatrix} k \\ k - v_g \\ k - v_g - v_h \end{pmatrix} \Big/ k \in \mathbb{N}. \quad (5)$$

This 2-D algorithm does not provide the vector in a particular order, and therefore, an algorithm to select and put in order the generating sequence is needed. This task must be done in accordance with some modulation strategy: capacitor balancing, commutation loss minimization, switch loss equalization, etc.

III. Implementation Design

A. Specifications

The main goal of this paper is to develop an optimal real-time implementation of both modulation algorithms for the NPC inverter in Fig. 2, which will be later implemented in an FPGA.

The switching frequency of the inverter can be selected in both implementations among 2.5, 5.0, 7.5, and 10 kHz by means of the input signal f_s. Vector reference is provided in the dq frame; hence, the input signals are the vector components v_d and v_q and the angle of the rotating frame θ. Finally, the modulator output signals are 12 trigger signals corresponding to the power switches of the NPC inverter. A symmetrical arrangement of the pulses into the switching cycle is carried out in order to get a good spectral output.

After analyzing similarities and differences between both algorithms, they have been organized as shown in the block diagrams in Fig. 3. The functional description of the blocks is detailed next. Bit number of the signals and hardware implementation concerning each block will be discussed in the next section.

B. Description of the 2-D SVPWM Algorithm Components

Fig. 3(a) shows the block diagram designed for the 2-D algorithm, which includes the following components: `freq`, `dq2gh`, `2DSVM`, `gh2abc`, `PWM3`, and `dead_time`.

1) Component `freq`*:* The user can select one of the four modulator switching frequencies available by means of the input signal f_s. The selected frequency defines the needed values for the switching period T and the dead time T_d. A dead time of 5% of the switching period was used.

2) Component `dq2gh`*:* This element carries out the following dq/gh transformation of the reference vector (v_d, v_q) from the rotational frame to the stationary frame

$$\begin{pmatrix} v_g \\ v_h \end{pmatrix} = \begin{pmatrix} \sqrt{3} & -1 \\ 0 & 2 \end{pmatrix} \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} \begin{pmatrix} v_d \\ v_q \end{pmatrix}. \quad (6)$$

3) Component `2DSVM`*:* This component is the main circuit of the system. It determines the nearest three vectors (v_{g1}, v_{h1}), (v_{g2}, v_{h2}), and (v_{g3}, v_{h3}) to the reference vector (v_g, v_h) in the gh frame and calculates their corresponding switching times [38].

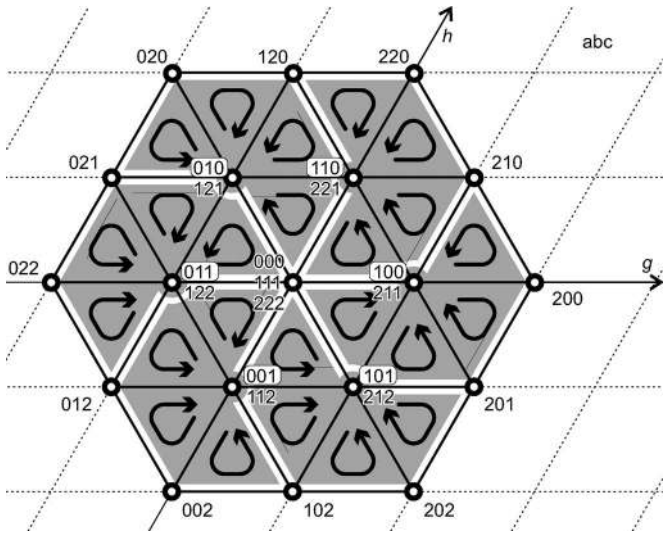

Fig. 4. Vector sequence selection.

4) Component `gh2abc`*:* This component transforms back the three nearest vectors from the gh to the abc frames. This is achieved by evaluating the expression (5). This expression has multiple solutions; therefore, the following algorithm to obtain the vector sequence was developed. The 24 triangular regions have been joined in the six highlighted groups shown in Fig. 4. The vector sequence starts with the boxed vector of the group, and it is tailored with adjacent vectors in order to minimize the switching number. That is, (v_{a1}, v_{b1}, v_{c1}) is the boxed vector, and vectors (v_{a2}, v_{b2}, v_{c2}) and (v_{a3}, v_{b3}, v_{c3}) are taken in accordance with the arrow inside the region. The following component `PWM3` generates a symmetrical sequence that starts and ends with the same vector; therefore, there are no additional switchings when the reference vector changes between triangular regions that belong to the same group, but it only adds two extra switchings when the reference changes to the next group. Therefore, this vector-selecting method minimizes the number of switchings when the reference vector stays inside a certain region as well as when it changes its region.

5) Component `PWM3`*:* It arranges the three vectors of the sequence in a symmetrical way into the switching period. The sequence generated is the following.

1st $(v_{a1}, v_{b1}, v_{c1}) \rightarrow T_1/4.$
2nd $(v_{a2}, v_{b2}, v_{c2}) \rightarrow T_2/2.$
3rd $(v_{a3}, v_{b3}, v_{c3}) \rightarrow T_3/2.$
4th $(v_{a1}+1, v_{b1}+1, v_{c1}+1) \rightarrow T_1/2.$
5th $(v_{a3}, v_{b3}, v_{c3}) \rightarrow T_3/2.$
6th $(v_{a2}, v_{b2}, v_{c2}) \rightarrow T_2/2.$
7th $(v_{a1}, v_{b1}, v_{c1}) \rightarrow T_1/4.$

After that, it generates the six PWM signals corresponding to each complementary pair of switches.

6) Component `dead_time`*:* Finally, this circuit generates the trigger signal for each power switch, introducing the proper dead time to give each insulated gate bipolar transistor (IGBT) enough time to switch off, before its complementary one is switched on. This is done by delaying the rising edges of the trigger signals the time T_d.

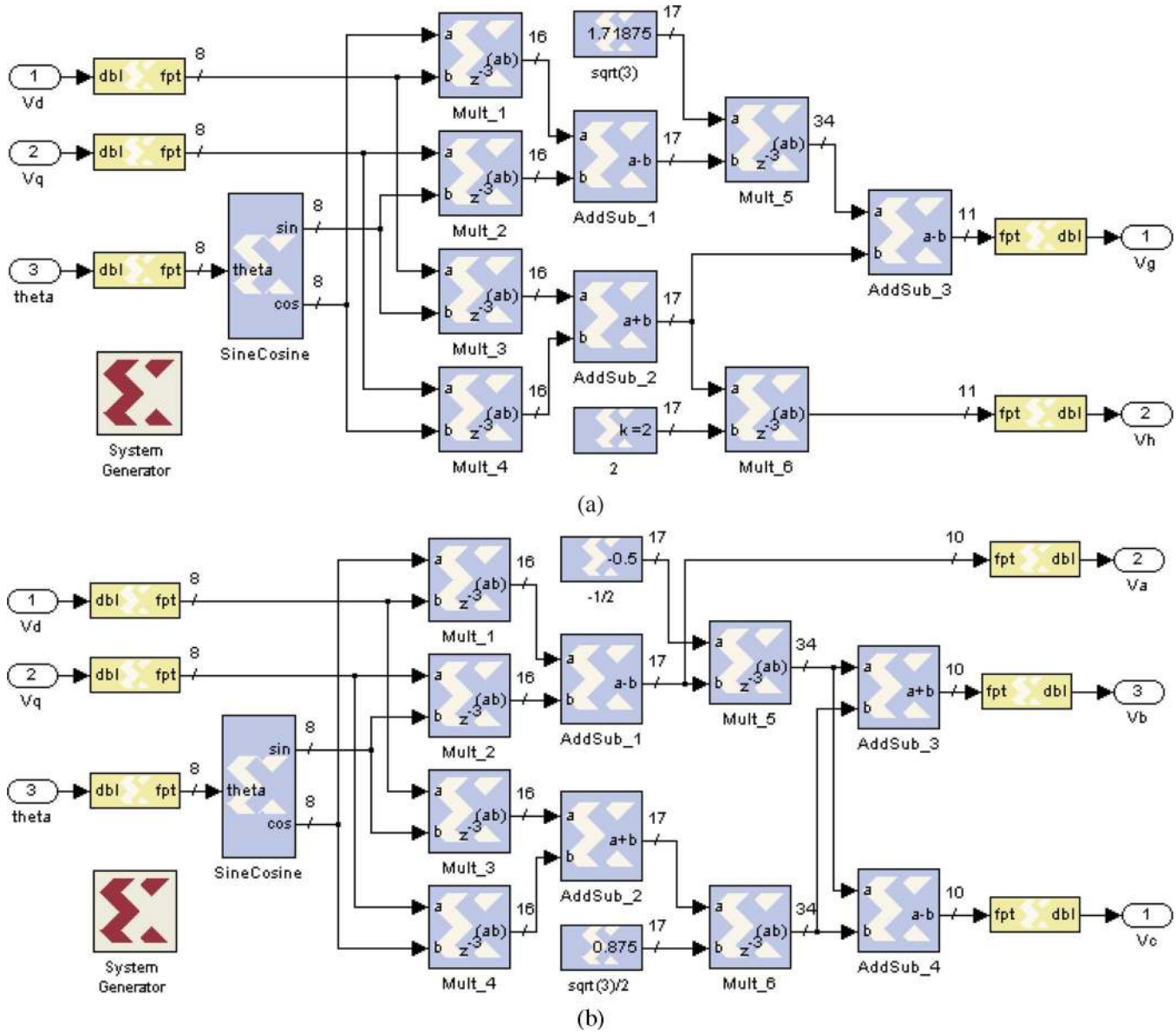

Fig. 5. Simulink implementation of the transformation components. (a) Component `dq2gh`. (b) Component `dq2abc`.

C. Description of the 3-D SVPWM Algorithm Components

Fig. 3(b) shows the simplified block diagram proposed for the 3-D algorithm. It is composed of the following main components: `freq`, `dq2abc`, `3DSVM`, `PWM4`, and `dead_time`.

1) Component `freq`*:* It is the same block used in the 2-D algorithm for the selection of the switching frequency that calculates the switching period T and the dead time T_d.

2) Component `dq2abc`*:* This component implements the following dq/abc transformation:

$$\begin{pmatrix} v_a \\ v_b \\ v_c \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ -\frac{1}{2} & \frac{\sqrt{3}}{2} \\ -\frac{1}{2} & -\frac{\sqrt{3}}{2} \end{pmatrix} \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} \begin{pmatrix} v_d \\ v_q \end{pmatrix}. \quad (7)$$

3) Component `3DSVM`*:* This component finds the four nearest vectors (v_{a1}, v_{b1}, v_{c1}), (v_{a2}, v_{b2}, v_{c2}), (v_{a3}, v_{b3}, v_{c3}), and (v_{a4}, v_{b4}, v_{c4}) to the reference vector and calculates their corresponding switching times. This problem has a unique solution, and the result sequence minimizes the switching number. Thus, no extra circuits like `gh2abc` are needed.

4) Component `PWM4`*:* This block is very similar to the `PWM3` but working with four input vectors instead of three. It arranges the vectors in a symmetrical way into the switching period and generates the six PWM signals corresponding to each complementary pair of IGBTs. The sequence generated is the following.

1st $(v_{a1}, v_{b1}, v_{c1}) \rightarrow T_1/2.$
2nd $(v_{a2}, v_{b2}, v_{c2}) \rightarrow T_2/2.$
3rd $(v_{a3}, v_{b3}, v_{c3}) \rightarrow T_3/2.$
4th $(v_{a4}, v_{b4}, v_{c4}) \rightarrow T_4.$
5th $(v_{a3}, v_{b3}, v_{c3}) \rightarrow T_3/2.$
6th $(v_{a2}, v_{b2}, v_{c2}) \rightarrow T_2/2.$
7th $(v_{a1}, v_{b1}, v_{c1}) \rightarrow T_1/2.$

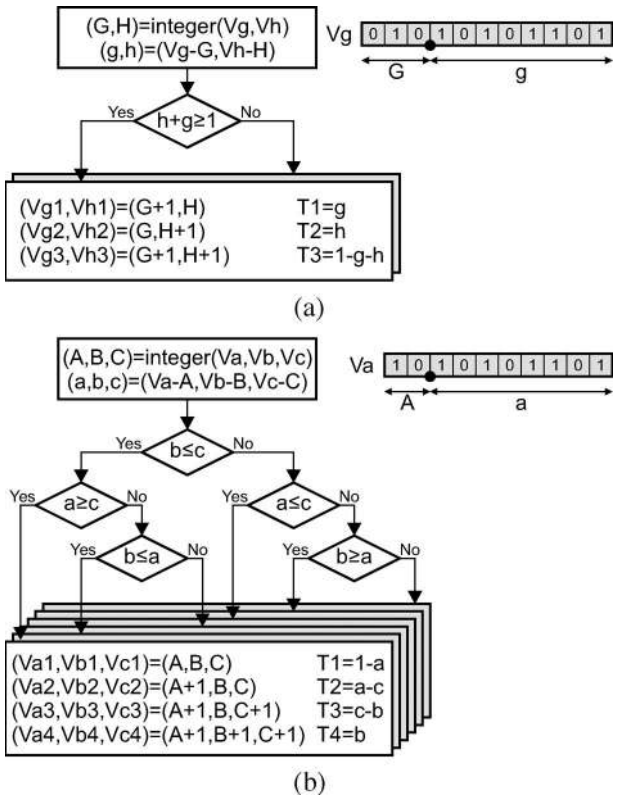

Fig. 6. Flow diagram of the VHDL description of the components `2DSVM` and `3DSVM`. (a) Component `2DSVM`. (b) Component `3DSVM`.

5) Component `dead_time`*:* This is the same component used in the 2-D algorithm to generate the 12 trigger signals inserting the corresponding dead times.

IV. Hardware Implementation

The Digilent S3 board was used to implement both algorithms. This board hosts a XC3S200 FPGA from Xilinx, which has 4.320 logic cells, each constituted by two 16×1 lookup tables (LUTs) and two flip-flops. This FPGA also has twelve 18×18 hardware multipliers, as well as twelve 18-kb block random access memories (BRAMs).

Although FPGAs allow a great degree of parallelism in the implementation of such algorithms, it is necessary to reduce the total amount of hardware needed, keeping the final cost at a reasonable point. Operations are done with a 10-MHz clock, which is enough frequency to achieve a real-time operation in this algorithm. The area of arithmetic circuits grows exponentially with the number of bits. A tradeoff between precision and area led to choose 8 b for the input variables: v_d, v_q, and θ. In the worst case, when the amplitude of the reference is equal to unity, the total error because of the bit truncation is 2.52%.

A. Component Implementation

The hardware description of the components of the algorithms shown in Fig. 3 has been hand coded in VHDL. An exception was made with the components `dq2gh` and `dq2abc`. These components have been developed in Simulink with the

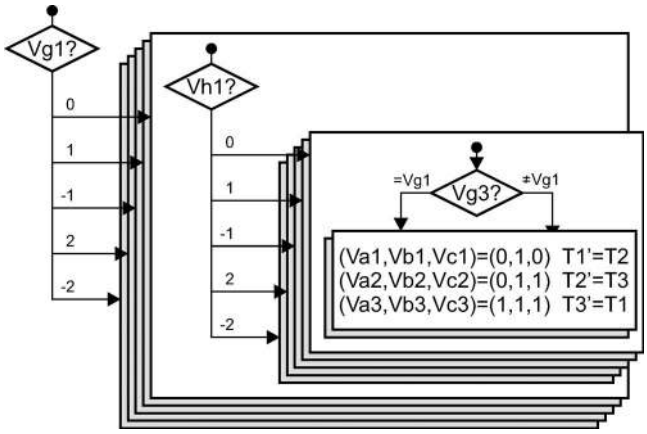

Fig. 7. Flow diagram of the VHDL description of the `gh2abc` component.

System Generator libraries provided by Xilinx. The VHDL code that describes these components has been automatically generated using the System Generator tool in order to produce a correct system implementation in a short design time.

1) Component `freq`*:* One of the four frequency and dead time pairs can be selected by means of the 2-b signal f_s. Signals T and T_d are integer numbers that express the switching period and the dead time in microseconds. Signal T is 10-b wide, which allows periods of up to 1023 μs, i.e., 977.5 Hz. T_d is a 6-b signal, which allows defining a dead time of up to 63 μs. The modulator switching period T and the corresponding dead time T_d are selected by means of a four-channel multiplexer.

2) Components `dq2gh` *and* `dq2abc`*:* They both implement transformations (6) and (7) from the rotating to the stationary frames. Fig. 5 shows the Simulink models used to describe the components and to generate their VHDL description files.

Internal arithmetic operations are done with adequate precision using fixed-point number representation and the two's complement format. The fractional part of the final result was rounded to 8 b in order to calculate the switching times with adequate accuracy in the components `2DSVM` and `3DSVM`. For an N-level inverter, in the 2-D algorithm, the integer part of the results takes values from $-(N-1)$ to $N-1$, whereas in the 3-D algorithm, it is in the range from 0 to $(N-1)$. Therefore, 3 and 2 b are necessary to represent the integer part, respectively. Consequently, the output signals in the 2-D case are 11-b wide, and the output signals in the 3-D case are 10-b wide. Both blocks could be used without any modification for an NPC inverter of up to five levels. If the number of levels grows, then more bits would be needed to represent properly these signals.

The sine and cosine operations have been implemented by means of a table stored in a memory. Sinusoidal waveform's 256 points have been stored with 8 b of resolution. In order to store these data, a 256×8 memory is needed; therefore, a BRAM was used to take advantage of the FPGA hardware resources.

3) Components `2DSVM` *and* `3DSVM`*:* Both elements have been implemented strictly following the algorithm descriptions given in [38] and [47] by means of simple arithmetic and comparison operations.

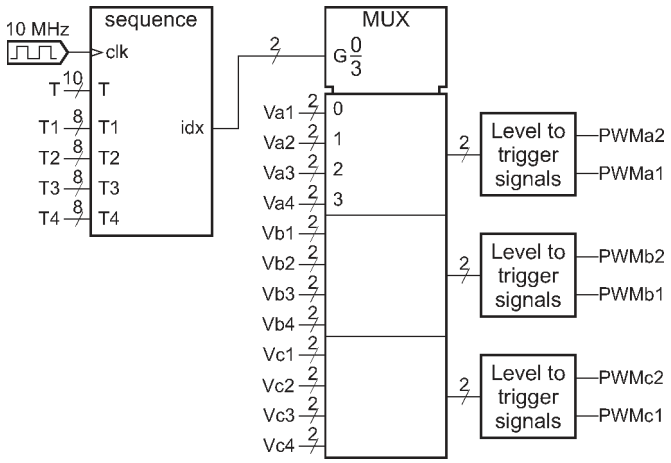

Fig. 8. Implementation diagram of the component PWM4.

The nearest vectors and unit switching times are calculated from the integer and the fractional parts of the reference, respectively. Therefore, the signals corresponding to the vector components have 3 b in the 2-D algorithm and 2 b in the 3-D algorithm, and the unit switching times are 8-b wide.

The integer and fractional parts of the references can be done by the proper bit extraction from the fixed point number. Corrections have been done in the case of negative numbers in the 2-D algorithm. Fig. 6 shows the flow diagrams of the components' VHDL description and the bits' extraction operations.

4) Component gh2abc*:* Fig. 7 shows the flow diagram utilized for the VHDL description of this block in accordance to Fig. 4. As shown previously, the output signals of the component have 2 b which are enough to represent the three levels of the inverter.

5) Components PWM3 *and* PWM4*:* Fig. 8 shows the component PWM4 implementation. The implementation of the component PWM3 is very similar. However, in this case, unit time T_4 is not needed, and the fourth vector is internally generated as

$$(v_{a4}, v_{b4}, v_{c4}) = (v_{a1} + 1, v_{b1} + 1, v_{c1} + 1). \tag{8}$$

The block sequence compares the switching time corresponding to each vector with the value of a counter to generate the vector index (idx) corresponding to each time interval. The 50-MHz master clock of the S3 board was divided by using the Digital Clock Manager of the FPGA in order to obtain a 10-kHz clock. That clock allows generating the PWM signals with a time precision of 0.1 μs. Due to the fact that vector times are 8-b wide, the maximum switching frequency available in this system is 39.2 kHz. Without the frequency division of the master clock, the maximum switching frequency could be increased five times, up to 196 kHz. The output of the multiplexer is the space vector that must be generated by the inverter at a particular time.

The level of each phase in the corresponding trigger signals is translated by the component Level to trigger signal. This translation depends on the inverter topology; therefore, this block must be redesigned if an inverter that is different from the NPC (with the same number of levels) is used [48]. It is not necessary to modify the rest of the components because both modulation algorithms do not depend on the topology.

TABLE I
RESOURCES SUMMARY

2D SVPWM implementation

Target Device	: xc3s200			
Number of Slice Flip Flops:	744	out of	3,840	19%
Number of occupied Slices:	1,547	out of	1,920	80%
Total Number 4 input LUTs:	2,702	out of	3,840	70%
Number of bonded IOBs:	41	out of	173	23%
IOB Flip Flops:	37			
Number of Block RAMs:	1	out of	12	8%
Number of MULT18X18s:	10	out of	12	83%
Number of GCLKs:	4	out of	8	50%
Number of Startups:	1	out of	1	100%
Total equivalent gate count for design:	133,126			

3D SVPWM implementation

Target Device	: xc3s200			
Number of Slice Flip Flops:	1,519	out of	3,840	39%
Number of occupied Slices:	1,438	out of	1,920	74%
Total Number 4 input LUTs:	2,120	out of	3,840	55%
Number of bonded IOBs:	41	out of	173	23%
IOB Flip Flops:	37			
Number of Block RAMs:	1	out of	12	8%
Number of MULT18X18s:	12	out of	12	100%
Number of GCLKs:	4	out of	8	50%
Number of Startups:	1	out of	1	100%
Total equivalent gate count for design:	145,551			

6) Component dead_time*:* This block generates the inverse of the trigger signals adding the corresponding dead time. It was implemented using an edge detector together with a counter working as a timer in order to delay the rising edges of the trigger signals.

B. Resources

All circuits were combined in a top file according to Fig. 3. At last, the whole system was synthesized and implemented in the XC3S200 FPGA through the use of the Xilinx Foundation ISE tools, which are specific for these tasks.

Table I compares the FPGA logic resources used to implement both algorithms. In our implementation, both algorithms use only one BRAM-dedicated memory to generate the sine and cosine functions, but the 2-D algorithm uses half flip-flops and two hardware multipliers less than the 3-D algorithm. This is because the first algorithm works with only three 2-D vectors, instead of the four 3-D vectors of the second algorithm. Although the 2-D algorithm works with few and small vectors, both implementations use a similar number of slices (logic blocks) and LUTs. This is due to the need for an extra task to select the generating vectors in the 2-D algorithm (component gh2abc). Consequently, the 3-D algorithm is easier to implement but uses more logical resources of the FPGA.

In order to extend the proposed implementation to a higher level inverter, many considerations have to be taken into account. The implementation of the component freq is independent of the inverter topology; therefore, it can be used without any modification. Components dq2gh, dq2abc, 2DSVM, and 3DSVM only need slight modifications in order to adapt the bit

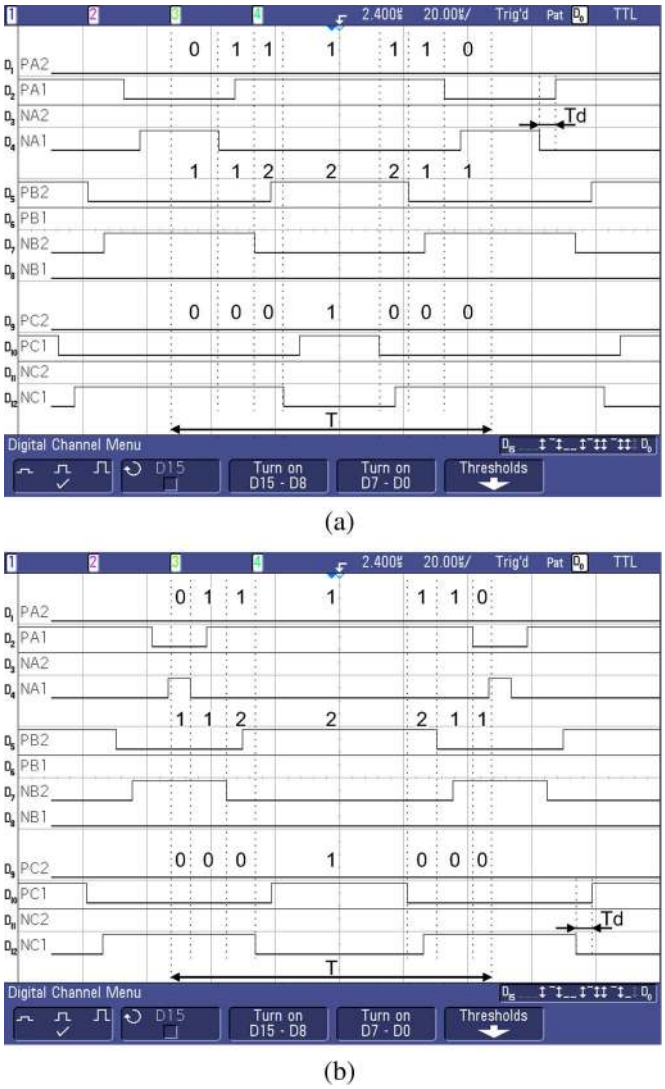

Fig. 9. Trigger signals. (a) 2-D SVPWM algorithm. (b) 3-D SVWPM algorithm.

number of the signals to the level number of the inverter. Resources used by those components grow in a stepped way with the number of levels. In fact, they could be directly used for a five-level topology. Components `PWM3`, `PWM4`, and `dead_time` have to be extended in order to generate additional PWM signals for extra IGBTs of the inverter. Therefore, resources used by the extended components grow almost linearly with the number of levels of the NPC inverter. Nevertheless, the component `gh2abc` was specifically designed for a three-level inverter, and therefore, it needs a major revision.

V. Experimental Results

Fig. 9 shows the FPGA outputs corresponding to the trigger signals for the 2-D and 3-D algorithms when $v_d = 0.6$, $v_q = 0$, and $\theta = 100°$. Vector sequence corresponding to this case and the rising edge delay because of the dead time can be seen in the figure.

Both FPGA implementations were tested with a three-level NPC inverter driving a star-connected induction motor. A

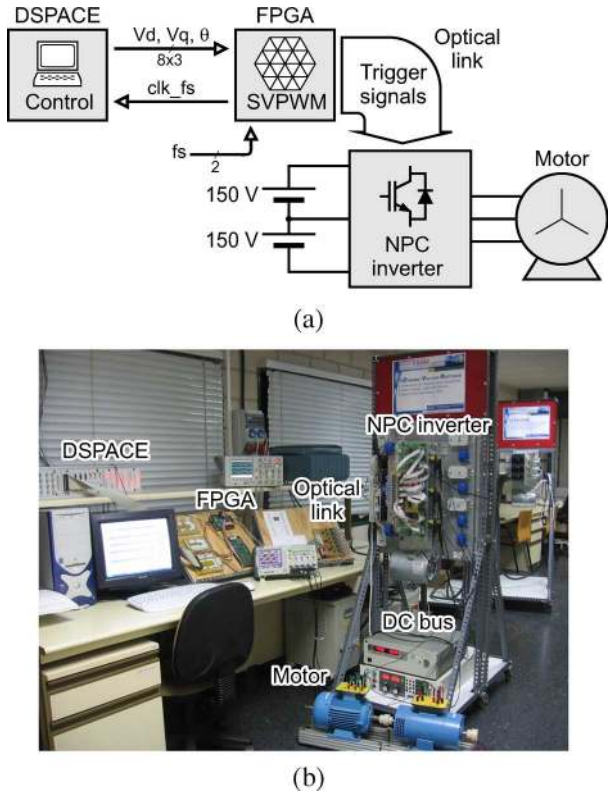

Fig. 10. Experimental test setup. (a) Diagram. (b) Photograph.

220/380-V, 1.420-r/min, and 1.35-kW rated motor was used. According to the motor characteristics, a dc bus of 300 V was selected. Fig. 10(a) shows a diagram of the experimental setup. The inverter is controlled in open loop in the rotating dq frame. The controller was implemented in a DSPACE DS1103 PPC Controller Board. The communication link between the DSPACE board and the FPGA uses a data bus of 3×8 b. Synchronization is made using the switching frequency clock. Every falling edge, the controller calculates a new set of references and writes them on the bus. New data are read by the FPGA on the following rising edge of the clock. Fig. 10(b) shows a photograph of the whole system used in the test, including the power converter, the FPGA board, and a personal computer.

Fig. 11 shows the line-to-line voltages and neutral-to-ground voltages measured with both implementations. As expected, the line-to-line voltages are very similar in both cases, and differences appear in the neutral-to-ground voltage. The filtered v_{ng} in the 3-D algorithm is constant and near to zero; therefore, homopolar component is not present. However, in the 2-D case, the filtered v_{ng} shows the presence of a homopolar component which allows better use of the dc bus.

VI. Conclusion

This paper compares the FPGA implementation of two different multilevel SVPWM algorithms: 2-D and 3-D.

The characteristics of both algorithms were studied in order to get an optimal hardware implementation. In the 2-D

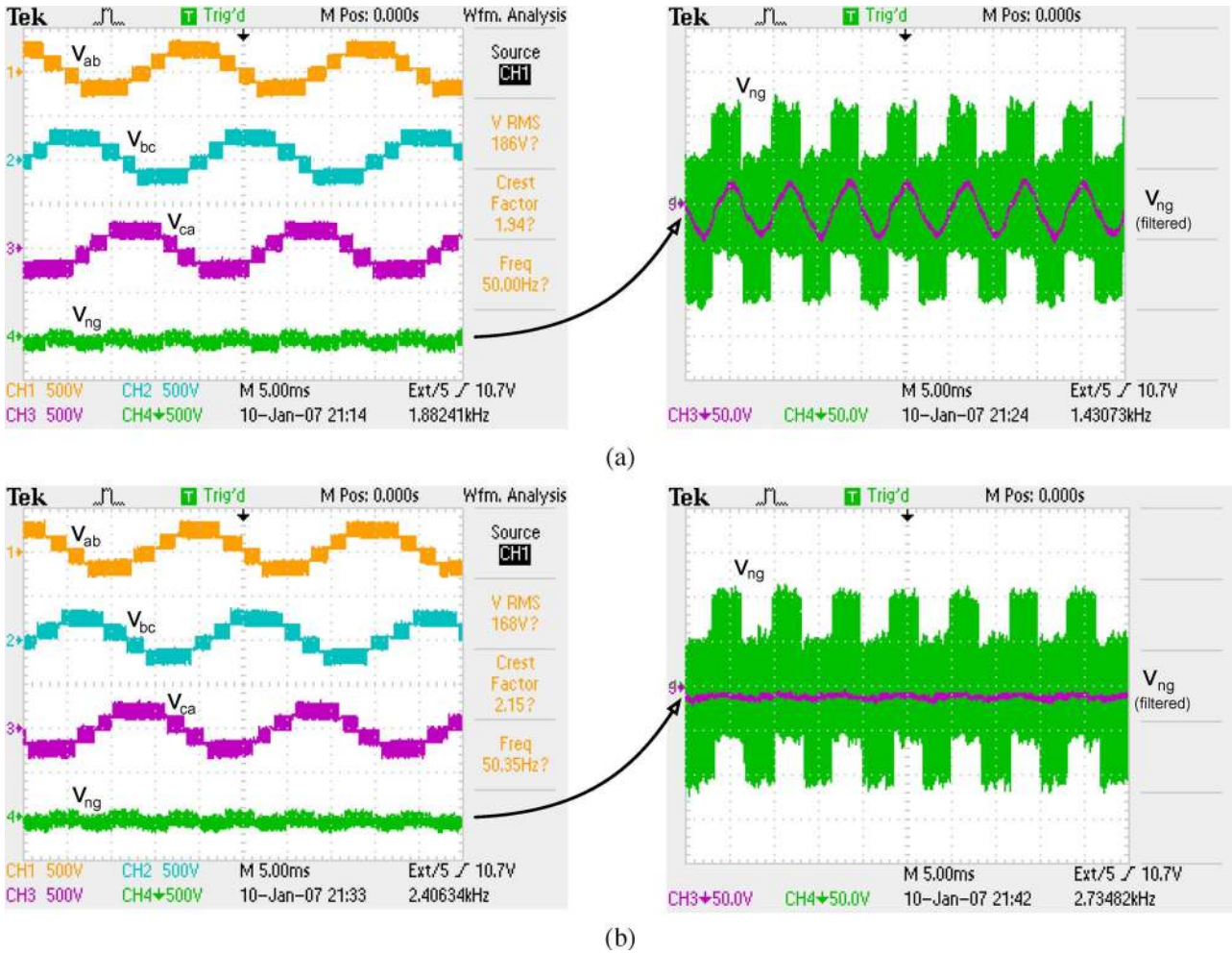

Fig. 11. Experimental waveforms during test. (a) 2-D SVPWM algorithm. (b) 3-D SVWPM algorithm.

algorithm, the need for an extra component to generate and arrange the final vector sequence was appointed. A particular solution for this component, which minimizes the switching looses in the inverter, was developed. Both algorithms were described in VHDL and implemented in an FPGA. It was shown that the 3-D algorithm needs fewer components, and it is easier to implement than its counterpart. In our implementation, the 2-D algorithm used less hardware resources of the FPGA. Finally, both algorithms were tested with a three-level neutral-clamped inverter.

REFERENCES

[1] A. Nabae, I. Takahashi, and H. Akagi, "A new neutral-point-clamped PWM inverter," *IEEE Trans. Ind. Appl.*, vol. IA-17, no. 5, pp. 518–523, Sep./Oct. 1981.

[2] B. Singh, B. N. Singh, A. Chandra, K. Al-Haddad, A. Pandey, and D. Kothari, "A review of three-phase improved power quality AC-DC converters," *IEEE Trans. Ind. Electron.*, vol. 51, no. 3, pp. 641–660, Jun. 2004.

[3] M. Marchesoni, P. Segarich, and E. Soressi, "A new control strategy for neutral-point-clamped active rectifiers," *IEEE Trans. Ind. Electron.*, vol. 52, no. 2, pp. 462–470, Apr. 2005.

[4] L. Yacoubi, K. Al-Haddad, L.-A. Dessaint, and F. Fnaiech, "Linear and nonlinear control techniques for a three-phase three-level NPC boost rectifier," *IEEE Trans. Ind. Electron.*, vol. 53, no. 6, pp. 1908–1918, Dec. 2006.

[5] S. Alepuz, S. Busquets-Monge, J. Bordonau, J. Gago, D. Gonzalez, and J. Balcells, "Interfacing renewable energy sources to the utility grid using a three-level inverter," *IEEE Trans. Ind. Electron.*, vol. 53, no. 5, pp. 1504–1511, Oct. 2006.

[6] R. Portillo, M. Prats, J. Leon, J. Sanchez, J. Carrasco, E. Galvan, and L. Franquelo, "Modeling strategy for back-to-back three-level converters applied to high-power wind turbines," *IEEE Trans. Ind. Electron.*, vol. 53, no. 5, pp. 1483–1491, Oct. 2006.

[7] N. Choi, J. Cho, and G. Cho, "A general circuit topology of multilevel inverter," in *Proc. 22nd Annu. IEEE PESC Rec.*, Cambridge, MA, Jun. 24–27, 1991, pp. 96–103.

[8] T. Meynard and H. Foch, "Multi-level conversion: High voltage choppers and voltage-source inverters," in *Proc 23rd Annu. IEEE PESC Rec.*, Jun. 29–Jul. 3, 1992, vol. 1, pp. 397–403.

[9] J. Lai and F. Z. Peng, "Multilevel converters—A new breed of power converters," *IEEE Trans. Ind. Appl.*, vol. 32, no. 3, pp. 509–517, May/Jun. 1996.

[10] J. Rodriguez, J. Lai, and F. Z. Peng, "Multilevel inverters: A survey of topologies, controls, and applications," *IEEE Trans. Ind. Electron.*, vol. 49, no. 4, pp. 724–738, Aug. 2002.

[11] J. J. Rodriguez-Andina, M. J. Moure, and M. D. Valdes, "Features, design tools, and application domains of FPGAs," *IEEE Trans. Ind. Electron.*, vol. 54, no. 4, pp. 1810–1823, Aug. 2007.

[12] K. Sridharan and T. Priya, "The design of a hardware accelerator for real-time complete visibility graph construction and efficient FPGA implementation," *IEEE Trans. Ind. Electron.*, vol. 52, no. 4, pp. 1185–1187, Aug. 2005.

[13] S. Sanchez-Solano, A. J. Cabrera, I. Baturone, F. J. Moreno-Velo, and M. Brox, "FPGA implementation of embedded fuzzy controllers for robotic applications," *IEEE Trans. Ind. Electron.*, vol. 54, no. 4, pp. 1937–1945, Aug. 2007.

[14] M.-W. Naouar, E. Monmasson, A. A. Naassani, I. Slama-Belkhodja, and N. Patin, "FPGA-based current controllers for AC machine drives—A review," *IEEE Trans. Ind. Electron.*, vol. 54, no. 4, pp. 1907–1925, Aug. 2007.
[15] E. Monmasson and M. N. Cirstea, "FPGA design methodology for industrial control systems—A review," *IEEE Trans. Ind. Electron.*, vol. 54, no. 4, pp. 1824–1842, Aug. 2007.
[16] M. Aime, G. Gateau, and T. A. Meynard, "Implementation of a peak-current-control algorithm within a field-programmable gate array," *IEEE Trans. Ind. Electron.*, vol. 54, no. 1, pp. 406–418, Feb. 2007.
[17] B. Lu, X. Wu, H. Figueroa, and A. Monti, "A low-cost real-time hardware-in-the-loop testing approach of power electronics controls," *IEEE Trans. Ind. Electron.*, vol. 54, no. 2, pp. 919–931, Apr. 2007.
[18] Y. F. Chan, M. Moallem, and W. Wang, "Design and implementation of modular FPGA-based PID controllers," *IEEE Trans. Ind. Electron.*, vol. 54, no. 4, pp. 1898–1906, Aug. 2007.
[19] Y.-A. Chapuis, L. Zhou, Y. Fukuta, Y. Mita, and H. Fujita, "FPGA-based decentralized control of arrayed MEMS for microrobotic application," *IEEE Trans. Ind. Electron.*, vol. 54, no. 4, pp. 1926–1936, Aug. 2007.
[20] M. N. Cirstea and A. Dinu, "A VHDL holistic modeling approach and FPGA implementation of a digital sensorless induction motor control scheme," *IEEE Trans. Ind. Electron.*, vol. 54, no. 4, pp. 1853–1864, Aug. 2007.
[21] J. Qin, C. Stroud, and F. F. Dai, "FPGA-based analog functional measurements for adaptive control in mixed-signal systems," *IEEE Trans. Ind. Electron.*, vol. 54, no. 4, pp. 1885–1897, Aug. 2007.
[22] H. Du, H. Qi, and X. Wang, "Comparative study of VLSI solutions to independent component analysis," *IEEE Trans. Ind. Electron.*, vol. 54, no. 1, pp. 548–558, Feb. 2007.
[23] H. Zhuang, K.-S. Low, and W.-Y. Yau, "A pulsed neural network with on-chip learning and its practical applications," *IEEE Trans. Ind. Electron.*, vol. 54, no. 1, pp. 34–42, Feb. 2007.
[24] A.-M. Lienhardt, G. Gateau, and T. A. Meynard, "Digital sliding-mode observer implementation using FPGA," *IEEE Trans. Ind. Electron.*, vol. 54, no. 4, pp. 1865–1875, Aug. 2007.
[25] S. Jung and S. Su Kim, "Hardware implementation of a real-time neural network controller with a DSP and an FPGA for nonlinear systems," *IEEE Trans. Ind. Electron.*, vol. 54, no. 1, pp. 265–271, Feb. 2007.
[26] R. Dubey, P. Agarwal, and M. K. Vasantha, "Programmable logic devices for motion control—A review," *IEEE Trans. Ind. Electron.*, vol. 54, no. 1, pp. 559–566, Feb. 2007.
[27] A. Polic, M. Curkovic, and K. Jezernik, "Matrix based design of the current source inverter with FPGA based implementation," in *Proc. 30th Annu. Conf. IEEE IECON*, Nov. 2–6, 2004, vol. 2, pp. 1640–1645.
[28] R. Garcia-Gil, J. Espi, E. Dede, and E. Maset, "An all-digital controlled AC-DC matrix converter with high-frequency isolation and power factor correction," in *Proc. IEEE Int. Symp. Ind. Electron.*, May 4–7, 2004, vol. 2, pp. 1075–1080.
[29] J. Van den Keybus, B. Bolsens, K. De Brabandere, J. Driesen, and R. Belmans, "DSP and FPGA based platform for rapid prototyping of power electronic converters and its application to a sampled-data three-phase dual-band hysteresis current controller," in *Proc. IEEE 33rd Annu. PESC*, Jun. 23–27, 2002, vol. 4, pp. 1722–1727.
[30] H. van der Broeck, H. Skudelny, and G. Stanke, "Analysis and realization of a pulsewidth modulator based on voltage space vectors," in *Proc. IEEE Ind. Appl. Conf.*, 1986, pp. 244–251.
[31] A. R. Beig, G. Narayanan, and V. T. Ranganathan, "Modified SVPWM algorithm for three level VSI with synchronized and symmetrical waveforms," *IEEE Trans. Ind. Electron.*, vol. 54, no. 1, pp. 486–494, Feb. 2007.
[32] A. Gupta and A. Khambadkone, "A space vector PWM scheme for multilevel inverters based on two-level space vector PWM," *IEEE Trans. Ind. Electron.*, vol. 53, no. 5, pp. 1631–1639, Oct. 2006.
[33] L. Franquelo, M. Prats, R. Portillo, J. Galvan, M. Perales, J. Carrasco, E. Diez, and J. Jimenez, "Three-dimensional space-vector modulation algorithm for four-leg multilevel converters using abc coordinates," *IEEE Trans. Ind. Electron.*, vol. 53, no. 2, pp. 458–466, Apr. 2006.
[34] D.-W. Chung, J.-S. Kim, and S.-K. Sul, "Unified voltage modulation technique for real-time three-phase power conversion," *IEEE Trans. Ind. Appl.*, vol. 34, no. 2, pp. 374–380, Mar./Apr. 1998.
[35] J.-H. Youm and B.-H. Kwon, "An effective software implementation of the space-vector modulation," *IEEE Trans. Ind. Electron.*, vol. 46, no. 4, pp. 866–868, Aug. 1999.
[36] M. Veenstra and A. Rufer, "PWM-control of multi-level voltage-source inverters," in *Proc. IEEE 31st Annu. PESC*, Galway, Ireland, Jun. 18–23, 2000, vol. 3, pp. 1387–1393.
[37] O. Alonso, L. Marroyo, and P. Sanchis, "A generalized methodology to calculate switching times and regions in SVPWM modulation of multilevel converters," in *Proc. 10th Eur. Conf. Multilevel Converters EPE*, 2001, pp. 920–925.
[38] N. Celanovic and D. Boroyevich, "A fast space-vector modulation algorithm for multilevel three-phase converters," *IEEE Trans. Ind. Appl.*, vol. 37, no. 2, pp. 637–641, Mar. 2001.
[39] M. M. Prats, L. G. Franquelo, J. I. Leon, R. Portillo, E. Galvan, and J. M. Carrasco, "A 3-D space vector modulation generalized algorithm for multilevel converters," *IEEE Power Electron. Lett.*, vol. 1, no. 4, pp. 110–114, Dec. 2003.
[40] Y.-Y. Tzou and H.-J. Hsu, "FPGA realization of space-vector PWM control IC for three-phase PWM inverters," *IEEE Trans. Power Electron.*, vol. 12, no. 6, pp. 953–963, Nov. 1997.
[41] S.-L. Jung, M.-Y. Chang, J.-Y. Jyang, L.-C. Yeh, and Y.-Y. Tzou, "Design and implementation of an FPGA-based control IC for AC-voltage regulation," *IEEE Trans. Power Electron.*, vol. 14, no. 3, pp. 522–532, May 1999.
[42] M. Malinowski, W. Kolomyjski, M. Kazmierkowski, and S. Stynski, "DSP/FPGA control for variable-speed wind turbines," in *Proc. 32nd Annu. Conf. IEEE IECON*, Paris, France, Nov. 7–10, 2006, pp. 5161–5166.
[43] *Matlab User Guide*, Mathworks, Natick, MA, 2005.
[44] *System Generator User Guide*, Xilinx, San Jose, CA, 2005.
[45] *Foundation ISE User Guide*, Xilinx, San Jose, CA, 2005.
[46] F. Z. Peng, J. Lai, J. W. McKeever, and J. VanCoevering, "A multilevel voltage-source inverter with separate DC sources for static VAr generation," *IEEE Trans. Ind. Appl.*, vol. 32, no. 5, pp. 1130–1138, Sep./Oct. 1996.
[47] M. M. Prats, L. G. Franquelo, J. I. Leon, R. Portillo, E. Galvan, and J. M. Carrasco, "A SVM-3D generalized algorithm for multilevel converters," in *Proc. 29th Annu. Conf. IEEE IECON*, Nov. 2003, vol. 1, pp. 24–29.
[48] O. López, J. Doval-Gandoy, C. M. Peñalver, J. Rey, and F. D. Freijedo, "Redundancy and basic switching rules in multilevel converters," *Int. Rev. Elect. Eng.*, pp. 66–73, Jan./Feb. 2006.

Óscar López (M'05) was born in Spain in 1975. He received the M.Sc. degree from the University of Vigo, Vigo, Spain, in 2001, where he has been working toward the Ph.D. degree in the Department of Electronic Technology since 2004.

He is currently an Assistant Professor with the University of Vigo. His research interests are in the areas of power switching converter technology.

Jacobo Álvarez was born in Vigo, Spain, in 1967. He received the engineering degree and the Ph.D. degree in electronics from the University of Vigo in 1991 and 1995, respectively.

Since 1997, he has been a Full Professor with the Department of Electronic Technology, University of Vigo. His main topics of interest include programmable logic devices and field programmable gate arrays architectures and design methods, which are applied to industrial control problems.

Jesús Doval-Gandoy (M'99) received the M.Sc. degree from the Polytechnic University of Madrid, Madrid, Spain, in 1991, and the Ph.D. degree from the University of Vigo, Vigo, Spain, in 1999.

From 1991 to 1994, he worked with industries. He is currently an Associate Professor with the Department of Electronic Technology, University of Vigo. His research interests are in the areas of ac power conversion.

Francisco D. Freijedo (M'07) was born in Spain in 1978. He received the M.Sc. degree in physics from the University of Santiago de Compostela, Santiago de Compostela, Spain, in 2002. Since 2003, he has been working toward the Ph.D. degree in the Department of Electronic Technology, University of Vigo, Vigo, Spain.

Since 2005, he has been an Assistant Professor with the University of Vigo. His research interests include power-quality problems, grid-connected switching converters, ac power conversion, and flexible ac transmission systems.

Andrés Nogueiras (M'99) was born in Rosario, Argentina, in 1967. He received the degree in industrial engineering and the Ph.D. degree (*cum laude*) in industrial engineering from the University of Vigo, Vigo, Spain, in 1994 and 2003, respectively.

He was a Research Assistant with the Applied Electronics Institute "Pedro Barrie de la Maza" in 1994. Since 1995, he has been an Assistant Professor with the Department of Electronic Technology, University of Vigo. His current research interests include power electronics systems, switched converter nonlinear modeling, applied reliable available maintainable and safe technologies, and teaching power electronics through Internet.

Alfonso Lago (M'94) was born in Lalín, Spain, in 1962. He received the Ph.D. degree in electrical engineering from the University of Vigo, Vigo, Spain, in 1994.

Since 1995, he has been an Associate Professor with the Department of Electronic Technology, University of Vigo. His main topics of interest include switching-mode power supplies and control applied to power converters.

Dr. Lago is a member of the IEEE Industrial Electronics Society.

Carlos M. Peñalver (M'92) was born in Spain in 1950. He received the M.Sc. and Ph.D. degrees in electrical engineering from the Universidad Politécnica de Madrid, Madrid, Spain, in 1977 and 1982, respectively.

He was an Associate Professor with the Universidad Politécnica de Madrid from 1977 to 1983, a Professor with the University of Santiago de Compostela, Santiago de Compostela, Spain, from 1984 to 1988, and a Full Professor with the Universidad de Oviedo, Oviedo, Spain in 1989. He has been a Full Professor with the Department of Electronic Technology, University of Vigo, Vigo, Spain, since 1990. His research interests are in the areas of power switching converter technology, flexible ac transmission systems, active power filters, and electrical machine drives.